DEBUGGING APPARATUS AND METHOD

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to a debugging apparatus and method, and more particularly, to a debugging apparatus that is capable of recognizing a data conversion state in a specific memory by observing a change and flow of a specific data of a memory and performing a debugging, and its method.

2. Description of the Background Art

In general, a debugging apparatus, detecting an error generated to a program inputted to a microprocessor, includes a host computer and a microprocessor unit (MPU).

In the debugging apparatus, when the host computer selects a specific address of a program memory in the MPU as a break point, the MPU monitors the processor while the processor is being operated, and when the specific address of the program memory is identical to the memory address selected as the break point, the MPU recognizes the selected memory address as a break point and discontinues the operation of the processor and the host computer observes a flow of a program and performs a debugging on the program.

Figure 1 is a schematic block diagram of the debugging apparatus in accordance with a conventional art.

As shown in Figure 1, the debugging apparatus includes: a host computer 1 for selecting a specific address of a memory as a break point and performing a

debugging; a debugger controller 2 for receiving a control command from the host computer 1 and outputting a break enable signal and a break point address; a processor core 4 being operated upon receiving a control signal from the debugger controller 2; a program memory 5 for storing a program of the processor core 4; a data memory 6 for storing a data of the processor core 4; and a break point sensing unit 3 for receiving the break enable signal and the break point address from the debugger controller 2, observing an address of the program memory 5 being executed in the processor core 4, and recognizing an address as a debugger break point and outputting a break signal to the debugger controller 2 if the address is sensed to be identical to the inputted break point address.

The operation of the debugging apparatus constructed as described above will now be explained.

When the processor is switched to a debugging mode, the host computer 1 outputs a processor core stop signal and a break point address to the debugger controller 2, for debugging.

The debugger controller 2 outputs a stop signal to the processor core 4 to suspend the processor core 4, and outputs a break point address and a break enable signal to the break point sensing unit 3.

When the break point sensing unit 3 stores a program address that the processor core 4 wants to suspend in the program memory, that is, stores a break point address, the host computer 1 operates the processor core 4 in the order of programs stored in the program memory 5.

While the processor core 4 is operated in the program order, the break point sensing unit 3 observes a program address outputted to the processor core 4.

Subsequently, when the break point sensing unit 3 detects an address of the same program memory 5 as that of the stored program address, it outputs a break signal to the debugger controller 2.

The debugger controller 2 suspends the operation of the processor core 4 according to the break signal inputted from the break point sensing unit 3 and shifts a debugging control right to the host computer 1, so that the host computer 1 may perform a debugging operation.

However, in the debugging method, since an operation is determined by an address of a specific program memory among the sequential programs, it is not possible to recognize a data flow according to a data memory. Thus, in case where an error occurs by a data, much time and expense is taken for debugging.

In addition, in case of creating a program, since the state of a data is hardly recognized, an error occurs in a data memory allocation amount. In addition, in case of reading out a wrong data in a calculating process, an erroneous result value is outputted only to cause a malfunction to a system.

SUMMARY OF THE INVENTION

Therefore, an object of the present invention is to provide a debugging apparatus and method that is capable of saving a time and an expense in performing a debugging operation by recognizing a data transition state in a specific data memory by observing a change state and flow of an address and a data of the specific data memory in a data memory.

To achieve these and other advantages and in accordance with the purpose of the present invention, as embodied and broadly described herein,

there is provided a debugging apparatus including: a processor core operated by a program stored in a program memory to read a data stored in a data memory or write a data; a debugger controller for performing a debugging on the processor core upon receipt of a command from a host computer and outputting a data break point address; and a memory break controller for observing an address of a data memory used by the processor core, recognizing an address as a break point address to output a break signal to the debugger controller and the processor core to suspend the operation of the processor core, when the address is sensed to be identical, and transmitting a corresponding address and data to the host computer through the debugger controller.

To achieve the above objects, there is further provided a debugging method including the steps of: outputting an address of a data memory to be observed, that is a break point address and a break enable signal, when a processor is switched to a debugging mode; storing the outputted break point address, and operating the processor in a general operation state; comparing the stored break point address and the address of the data memory currently used by the processor core, while the process is being operated; outputting a break signal to suspend the process core, if the address of the data memory currently used by the processor core and the stored break point address are identical to each other; and suspending the processor core by the outputted break signal and switching the processor to a debugging mode to debug the program.

The foregoing and other objects, features, aspects and advantages of the present invention will become more apparent from the following detailed description of the present invention when taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are included to provide a further understanding of the invention and are incorporated in and constitute a part of this specification, illustrate embodiments of the invention and together with the description serve to explain the principles of the invention.

In the drawings:

Figure 1 is a schematic block diagram showing a debugging apparatus in accordance with a conventional art;

Figure 2 is a schematic block diagram showing a debugging apparatus in accordance with a preferred embodiment of the present invention;

Figure 3 is a detailed block diagram showing a memory break controller of Figure 2 in accordance with the preferred embodiment of the present invention;

Figure 4 is a schematic block diagram showing the structure of a memory break control register of Figure 3 in accordance with the preferred embodiment of the present invention; and

Figure 5 is a flow chart of a debugging method in accordance with the preferred embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Reference will now be made in detail to the preferred embodiments of the present invention, examples of which are illustrated in the accompanying drawings.

Figure 2 is a schematic block diagram showing a debugging apparatus in accordance with a preferred embodiment of the present invention.

As shown in Figure 2, a debugging apparatus of the present invention includes a host computer 10 for assigning a break point address and controlling a debugging operation; a debugger controller 20 for outputting a control signal to control a processor outputted from the host computer 10, a break point address and a break enable signal; a program memory 60 for storing a program to operate a processor core 40 (to be described); a processor core 40 being operated by the control signal outputted from the debugger controller 20 and the program stored in the program memory 60, and outputting a data generated accordingly; a data memory 70 for storing a data outputted from the processor core 40; a break point sensing unit 30 for observing an address of the program memory 60 used by the processor core 40, and recognizing an address as a debugger break point and transmitting a break signal to the debugger controller 20, if the address is sensed to be identical to the break point address outputted from the debugger controller 20; and a memory break controller 50 for observing an address and a data of the data memory 70 used by the processor core 40, and transmitting a break signal to the debugger controller 20 and the processor core 40 to suspend an operation of the processor core 40, transmitting the sensed address and its corresponding data to the host computer 10 through the debugger controller 20, and activating an operation of the processor core 40 to transmit the address and data of the data memory used in the processor core 40 to the host computer 10 through the debugger controller 20 until a break signal is outputted again, when the same address as the break address outputted from the debugger controller 20 is sensed.

The host computer 10 recognizes a data flow and change according to the address and the data outputted from the memory break controller 50.

The construction of the memory break controller of the debugging

apparatus will now be described with reference to Figure 3.

Figure 3 is a detailed block diagram showing a memory break controller of Figure 2 in accordance with the preferred embodiment of the present invention.

As shown in Figure 3, the memory break controller 50 includes a memory break control register 51 being activated by a control signal and a break point address signal outputted from the debugger controller 20 and observing a data change of the processor core 40; an address register (AR) 52 for storing the break point address outputted from the memory break control register 51; an address comparator (AC) 53 for comparing the address currently outputted from the data memory 70 and the break point address stored in the address register 52; a data register (DR) 54 for storing a data of the break point address stored in the address register 52; and a data comparator (DC) 55 for comparing the data of the address outputted from the data memory 70 and the data value of the break point address stored in the data register 54.

The construction of the memory break control register 51 will now be described with reference to Figure 4.

Figure 4 is a schematic block diagram showing the structure of a memory break control register of Figure 3 in accordance with the preferred embodiment of the present invention.

As shown in Figure 4, the memory break control register 51 includes: a memory break enable flag (MBEF) 51-1 for activating the memory break controller 50; a data check flag (DCF) 51-2 for sensing an address of the data memory which is identical to the break point address stored in the address register 52 and being enabled when the data of the corresponding address is outputted; and an address trace check flag (ACF) 51-3 assigning an initial break point address,

being enabled when a content of the break point address is read during the processing procedure, and outputting the content of the data and the addresses of every memory read or written from or into the processor until the content of the break address is updated.

The operation of the debugging operation constructed as described above will now be explained with reference to Figure 5.

Figure 5 is a flow chart of a debugging method in accordance with the preferred embodiment of the present invention.

As shown in Figure 5, while the processor in the host computer 10 is being operation, when it is switched to a debugger mode, the host computer 10 disables the debugger mode, so that the host computer sets a start position of a debugger mode program as a start position of the processor and outputs an initialization control signal and an address of the data memory to be observed, that is, a break point address, to the debugger controller 20.

According to the control signal and the break point address outputted from the host computer 10, the debugger controller 20 outputs the control signal, the break point address and the break enable signal to the memory break controller 50 and the break point sensing unit 30.

The break point sensing unit 30 receives and stores the break enable signal and the break point address.

According to the inputted break enable signal, the memory break control register 51 of the memory break controller 50 enables a memory break enable flag as '1' and disables a data check flag and an address track flag as '0' (step S10), thereby initializing the processor.

The address register 52 of the memory break controller 50 stores the

inputted break point address (step S11).

After completing the process, the host computer 10 operates the processor core 40 according to programs stored in the program memory 60.

When the processor core 40 is operated according to the program sequentially stored in the program memory, the break point sensing unit 30 monitors whether the break point address stored in the processor core 40 and the address of the program memory 60 used by the processor core 40 are identical to each other.

The address comparator 53 of the memory break controller 50 compares the address of the data memory 70 used by the processor core 40 and the break point address (step S12).

Upon comparison, if the address of the data memory 70 read from the processor core 40 and the break point address stored in the address register 52 are identical to each other, that is, if the address comparator 53 is enabled, the address comparator 53 outputs an accord signal to the debugger controller 20.

The debugger controller 20 judges whether the processor core 40 is reading a data for the corresponding address of the data memory 70 or writing a data to the address of the data memory 70 (step S14).

According to judgement result, in case that the processor core 40 writes a data in the data memory 70, the debugger controller 20 enables a data check flag (51-2) of the memory break control register 51 (step S18) and outputs a break signal for suspending execution of the program of the processor core 40 to the the processor core 40 (step S20), so as to discontinue the operation of the processor core.

Meanwhile, in case that the processor core 40 reads a data of an address

of the corresponding data memory 70, the debugger controller 20 enables an address trace check flag (ACF) 51-3 and a data check flag (CDF) 51-2 of the memory break control register 51 (step S15).

The data of a specific address of the data memory (70) read by the processor core 40 has an arithmetic and logical operation relation with a data value of a difference address or a correlation with a specific address of the data memory 70.

Accordingly, in case that a data of a specific address is read, when the address trace check flag 51-3 and the data check flag 51-2 are enabled, the address comparator 53 compares the specific address and the break point address (step S16).

When the break point address stored in the address register 52 and the address of the data memory 70 are identical to each other, the address comparator 53 transmits an accord signal to the debugger controller 30.

If, however, the break point address and the address of the data memory are not identical to each other, that is, if the value (AC) of the address comparator 53 is not '1', since the break point address and the address to be observed are different arithmetically and logically, the address comparator 53 and the data comparator 55 transmit the sensed address and data of the corresponding data memory to the host computer 10 through the debugger controller 20, for a debugging operation. (step S17).

Meanwhile, when the address comparator 53 is enabled, it means that the previously read address to be observed is used, the debugger controller 20 determines whether the processor core 40 is reading a data of the address to be observed of the data memory 70 or a data value is writing into the address to be

observed (step S19).

In case that a data of the address to be observed of the data memory 70 is being read, since it is the previously read address to be observed, it returns to the step in which the address comparator 53 of the memory break controller 50 compares the read address and the break point address and determines whether the read address is identical to the break point address stored in the address register 52 (step S16).

Meanwhile, in case that a data value is being written in a address to be observed, since it means that a result value according to an arithmetic operation is being written in the previously read address to be observed, the memory break controller 50 outputs a break signal to the debugger controller 20 and the processor core 40 (step S20), in order to discontinue the processor core 40.

Upon receiving the break signal, the debugger controller 20 outputs a signal allowing the host computer 10 to start a debugging operation. The host computer 10 performs a debugging operation according to the outputted signal.

Meanwhile, the address to be observed and the address comparison result of the data memory currently used by the processor core 40 are different to each other (step S12), the memory break controller 50 outputs the address and data of the data memory 70 currently read by the processor core 40 to the host computer 10.

Subsequently, the address comparator 53 repeats from the step for comparing the address of the next data memory 70 used by the processor core 40.

The operation of the debugging apparatus as described above will now be explained in detail by taking the following program as an example.

A0 = a ---------- (1)

A1 = b ---------- (2)

A2 = c ---------- (3)

A2 = A2 + (A1 * A2) --------- (4)

wherein A0, A1 and A2 are addresses of the data memory.

In the program constructed as described above, the host computer 10 designates a address to be observed as A2 and executes the program in a debug mode.

The address comparator 53 executes a debugging while comparing A2, the address to be observed, and the currently used address, and when the address comparator 53 reads a data value 'c' corresponding to A2, it senses A2 as an address to be observed.

When the address comparator 53 senses A2, the memory break control register 51 outputs a break signal to the processor core 40 in order to discontinue the operation, and determines whether A2 of formula (3) is for reading or writing.

Upon determining, if A2 is for reading, the memory break control register 51 transmits the A2 address and the data value 'c' to the host computer 10.

In case of executing continuously, the memory break controller 50 reads A2, so that addresses of every memory related until the next A2 value is written and their contents are outputted. That is, the address comparator 53 continuously executes the processor, stops at A2 of the formula (4), and compares A0, A1 and A2 with A2, the address to be observed.

After the address comparator 53 first compares A0 and A1 with the address to be observed A2, since A0 and A1 are not identical to the address to be observed, the address comparator 53 transmits a corresponding address and data value to the host computer 10 and compares A2 of the right side of the formula (4)

with the address to be observed.

Wince A2 is identical to the address to be observed, the memory break control register 51 determines whether it is reading or writing.

In this respect, A2 presented in the formula (3) is an address which has been read in the previous process, it goes to the next address.

Subsequently, after the right side of A2 of the formula (4) is compared, the left side is compared. In this respect, after A2 is sensed, it is determined whether it is reading or writing.

Since A2 of the formula (4) is for writing, the memory break controller 50 outputs a break signal to the debugger controller 20 and the processor core 40 to suspend the operation of the processor core 40.

The debugger controller 20 outputs a debugging mode switch signal to the host computer 10 by the inputted break signal.

Upon receiving the debugging mode switch signal from the debugger controller 20, the host computer 10 operates A2 by the previously transmitted address and data and updates A2 of the data memory to the operated value.

In case that a data constructed with arrangement in the address register 52 is observed, the host computer 10 designates a break point address as an uppermost address and a lowermost address of an arrangement of the address register 52 of the memory break controller 50.

The address register 52 stores the uppermost address and lowermost address of the arrangement of the data register 54, and sets a space for storing a data of the arrangement of the data register 54.

The address comparator 53 compares the range of the arrangement stored in the address register and the break point address and senses an address

of the data memory corresponding to the range of the arrangement. The following operation is performed in the same manner as the case of observing one data.

As so far described, the debugging apparatus and method of the present invention has the following advantages.

That is, for example, in a debugging operation, since an address and a data of a specific data memory are monitored to recognize a data flow and change of the specific address, an error that an erroneous calculation is inputted a during processing or a data memory is erroneously assigned is quickly sensed. Thus, a time and an expense for a debugging operation can be much saved.

In addition, by adding a data debugging method to the conventional program debugging method, a program development environment is set similar to an environment that the processor is substantially operated. Thus, a time and an expense for developing a program can be also saved.

As the present invention may be embodied in several forms without departing from the spirit or essential characteristics thereof, it should also be understood that the above-described embodiments are not limited by any of the details of the foregoing description, unless otherwise specified, but rather should be construed broadly within its spirit and scope as defined in the appended claims, and therefore all changes and modifications that fall within the meets and bounds of the claims, or equivalence of such meets and bounds are therefore intended to be embraced by the appended claims.